

Greedy local search

```
procedure GenSAT( $\Sigma$ )  
  for  $i = 1$  to  $Max\text{-}tries$   
     $T = initial(\Sigma)$   
    for  $j = 1$  to  $Max\text{-}flips$   
      if  $T$  satisfies  $\Sigma$  then return  $T$   
      else    $Poss\text{-}flips = hill\text{-}climb(\Sigma, T)$   
               $V = pick(Poss\text{-}flips)$   
               $T = T$  with  $V$ 's assignment flipped  
      endif  
    endfor  
  endfor  
  return "No satisfying assignment found"  
end GenSAT
```

GSAT

- *initial* : returns a randomly generated truth assignment
- *hill-climb* : returns variables whose truth assignment if flipped give the greatest increase in the number of satisfied clauses
 - called the *score* of a variable
 - greatest increase can be zero (*sideways* moves) or negative (*uphill* moves)
- *pick* : returns one of the variables at random

GSAT performance

- Table 1 from Selman *et al.* 1992

Sideways moves

- Table 4 from Selman *et al.* 1992

Variants of GSAT

- Is greediness important?
- Is randomness important?
 - in picking between equally good variables?
 - in picking the initial assignment for each try?
- Is memory useful?

Greediness

- TSAT: returns variables that increase the score the *least*, or if no variables increase the score then all sideways moves, or if no sideways moves then all moves
- TSAT performance is comparable to GSAT
- ...but *hill-climbing* is important

Randomness

- Is randomness important in picking variables?
 - DSAT: picks between equally good variables in a *deterministic* but *fair* way (variables are picked in a cyclic order)
 - DSAT outperforms GSAT
 - ...but *fairness* is important
- Is randomness important in the generating initial assignments?
 - VSAT: generates initial assignments in a deterministic order, but maximizes variance between successive tries
 - VSAT comparable to GSAT
 - ...but *variance* between successive tries is important

Memory

- HSAT: picks the variable flipped longest ago in this try
- HSAT significantly outperforms GSAT and DSAT
- Tabu lists in combinatorial optimization

Percent of problems solved vs total flips

- Figure 1 from Gent and Walsh 1993

Optimal value of Max-flips

- Figure 2 from Gent and Walsh 1993

GSAT with random walk

- With probability p , pick a variable occurring in some unsatisfied clause and flip its truth assignment
- With probability $1 - p$, follow the standard GSAT scheme, i.e., make the best possible local move

Comparing noise strategies

- Table 1 from Selman, Kautz, and Cohen 1994

WalkSAT

- *initial* : same as GSAT, i.e., random assignment
- *hill-climb* :
 - pick an *unsatisfied* clause
 - with probability p return variables with smallest *break count*
 - *break count* : # of clauses unsatisfied by flip
 - with probability $1 - p$ return all variables in clause
- *pick* : same as GSAT, i.e., pick randomly
- p must be tuned, but 0.5 works well in most cases
- *Max-flips* is usually $O(N^2)$, but *Max-tries* is usually 10 - 20.

Davis-Putnam

function DP(Σ , \mathbf{P})

Unit propagate Σ

if a contradiction is discovered **then return** *false*

else if all variables are valued **then return** *true*

else

Let x be some unvalued variable

return DP($\Sigma \cup \{x\}$, \mathbf{P}) **or** DP($\Sigma \cup \{\neg x\}$, \mathbf{P})

endif

end DP

Branch variable selection heuristics

Key idea: *Prefer variables that would cause a large number of unit propagations*

- Estimate the number of unit propagations caused by assigning a variable *true* and assigning it *false*
- Combine the two estimates to give a *score* for the variable

MOM's heuristic

- Select variables that have the *Maximum Occurrence* in clauses of *Minimum* length
- For each variable, x , *incrementally* keep track of
 - $pc(x)$: number of *binary* clauses in which x occurs *positively*
 - $nc(x)$: number of *binary* clauses in which x occurs *negatively*
- Score variables by combining the nc and pc values
 - *e.g.*, $score(x) = pc(x) * nc(x) * 1024 + pc(x) + nc(x) + 1$

Unit propagation heuristic

- For each variable, x , explicitly compute
 - $up+(x)$: actual number of unit propagations that result from assigning x to be true
 - $up-(x)$: actual number of unit propagations that result from assigning x to be false
- Combine scores as before

Combined heuristics

- Use MOM's heuristic to rank variables
- Compute unit propagation heuristic for top k variables
 - *e.g.*, $k = N - 21 * vars_valued$
 - Improved variations from [Li & Anbulagan IJCAI 97]